A

PROJECT REPORT

ON

“WORKOUT MANAGEMENT SYSTEM”

SUBMITTED BY:

Mr. Abhijeet Kanawade(2124UCEM1007)

SUBJECT: Programming And Problem-Solving Using C++

Under the guidance of

Miss. ISHWARI TIRSE

Department of Computer Science and Engineering

Sanjivani Rural Education Society’s

SANJIVANI UNIVERSITY

KOPARGAON – 423603, DIST: AHMEDNAGAR

2024-2025

INDEX

INTRODUCTION

This code is a basic fitness workout planner created in C++. It allows users to keep track of their strength workouts by adding new ones and viewing the list of workouts they've created. The program uses two main classes: Workout, which stores the name and duration of each workout, and Strength, which adds the number of sets for strength training exercises.

Users can interact with the program through a simple menu system. They sets, or they can view all their workouts. The program also ensures that it handles memory correctly by cleaning up any allocated resources when the user exits, making it a practical tool for anyone looking to organize their fitness routine. Overall, it's designed to be user-friendly and efficient, helping users stay on top of their workout plans.

Some of key features of system include-

1. Workout Management-The program allows users to add, view, edit, and delete workout entries, facilitating effective workout tracking.

2. Age Filtering-Users can filter and display workouts based on a specific age, helping to tailor fitness plans to different age groups.

3. Data Validation-The code includes checks for valid indices when viewing, editing, or deleting workouts, enhancing the robustness of user input handling.

4. Workout Count Tracking- The program keeps track of the total number of workouts added, providing users with insights into their fitness activities.

Objectives

This program is designed to help users track their strength workouts. It allows users to input details like the workout name, duration, and the number of sets for each workout. These details are stored in an array, with a limit of 10 workouts. Users can either add new workouts, view existing ones, or exit the program via a simple menu system. The workouts are represented by objects of the `Strength` class, which extends the general `Workout` class, allowing for better organization and management of workout data.

To manage memory, the program dynamically allocates memory for each workout when it's added and ensures that memory is properly freed when the program ends. The program also checks if users attempt to add more than the allowed 10 workouts and handles invalid menu options. Through this setup, the program combines basic concepts of object-oriented programming, like inheritance and polymorphism, with practical functionality for workout tracking.

Scope of Project is-

1. Workout Management-The program provides functionalities to add, view, edit, delete, and filter workouts based on age.

2. User Interaction- It uses a text-based menu system for user input, facilitating easy navigation through various features.

3. Data Handling- It utilizes a fixed-size array to store workout information, demonstrating basic data structure usage in C++.

Methodology

1. Object-Oriented Design-Utilizes a Workout class to encapsulate workout properties and behaviors.

2. User Interaction-Provides a console menu for user input to manage workouts, including adding, viewing, editing, and deleting entries.

3. Data Handling-Uses a fixed-size array to store workouts, limiting the total number to 10.

4. Loop Control-Uses a do-while loop to repeatedly display the menu until the user chooses to exit.

Expected outcomes

Aims of the Code/System:

1. Workout Management-To allow users to add and view strength workouts, managing essential details like name, duration, and sets.
2. User Interaction- To provide a simple menu-driven interface for interaction with options to add a workout, view workouts, or exit.
3. Memory Management- Ensure that memory is dynamically allocated for each new workout and properly cleaned up when the program exits.

Expected Benefits:

1. Easy Workout Tracking-Users can track multiple workouts and view them in a clear, organized format.
2. Simple User Interface- The menu system makes it straightforward for users to interact with the program, even if they are not tech-savvy.
3. Personalization-Users can input workout details like the workout name, duration, and number of sets, customizing their workout plan.
4. Memory Efficiency-Proper memory management prevents memory leaks by ensuring that dynamically allocated memory is freed when the program exits.

Expected Outcomes:

1. Workouts Added Successfully-Users will be able to add up to 10 strength workouts with their details.
2. Workouts Displayed Correctly-Upon selecting the option to view workouts, all previously added workouts will be listed with their details.

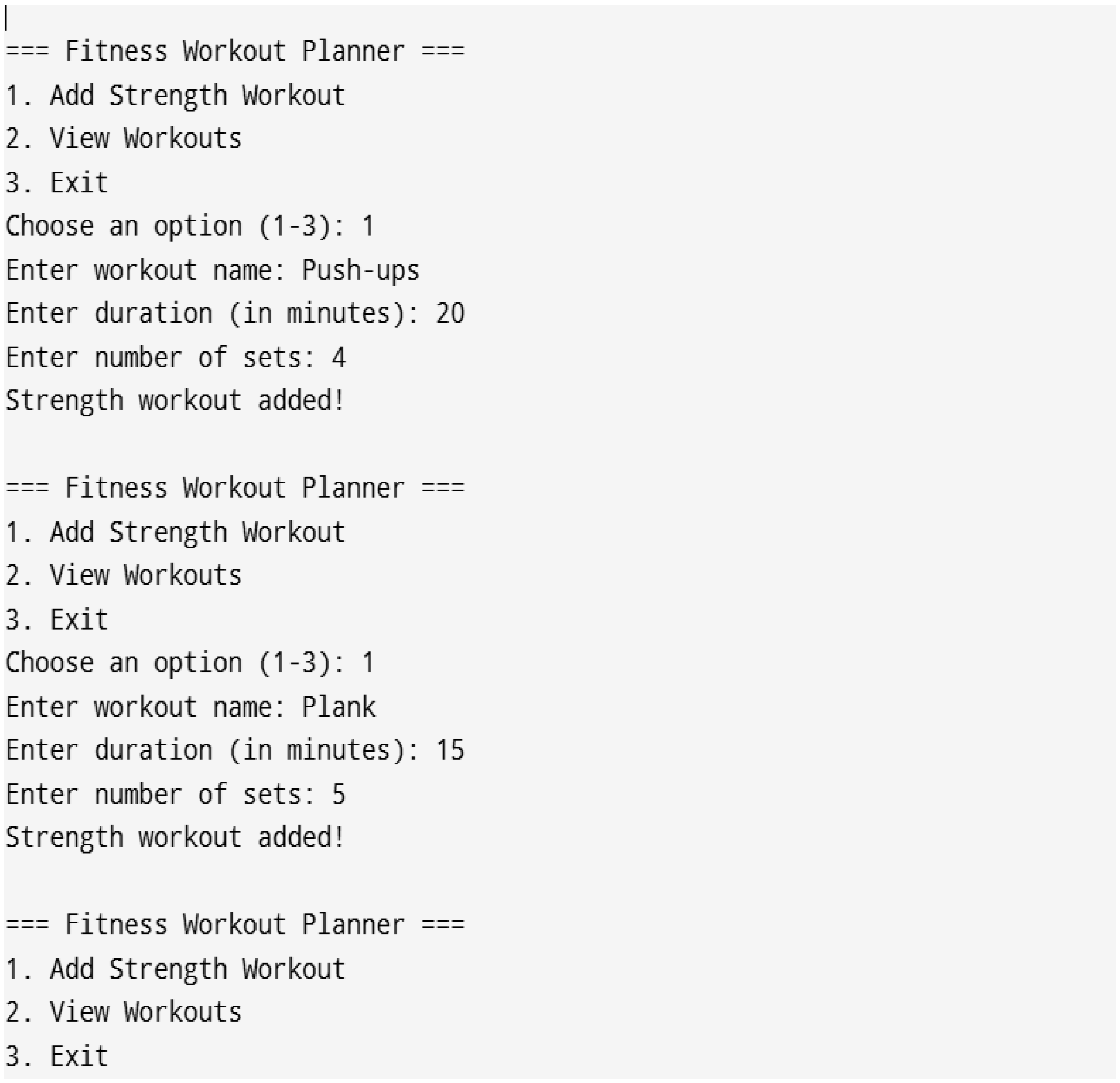
3.Error Handling- If the user tries to add more than 10 workouts, they will be notified that maximum limit has been reached.

CODE

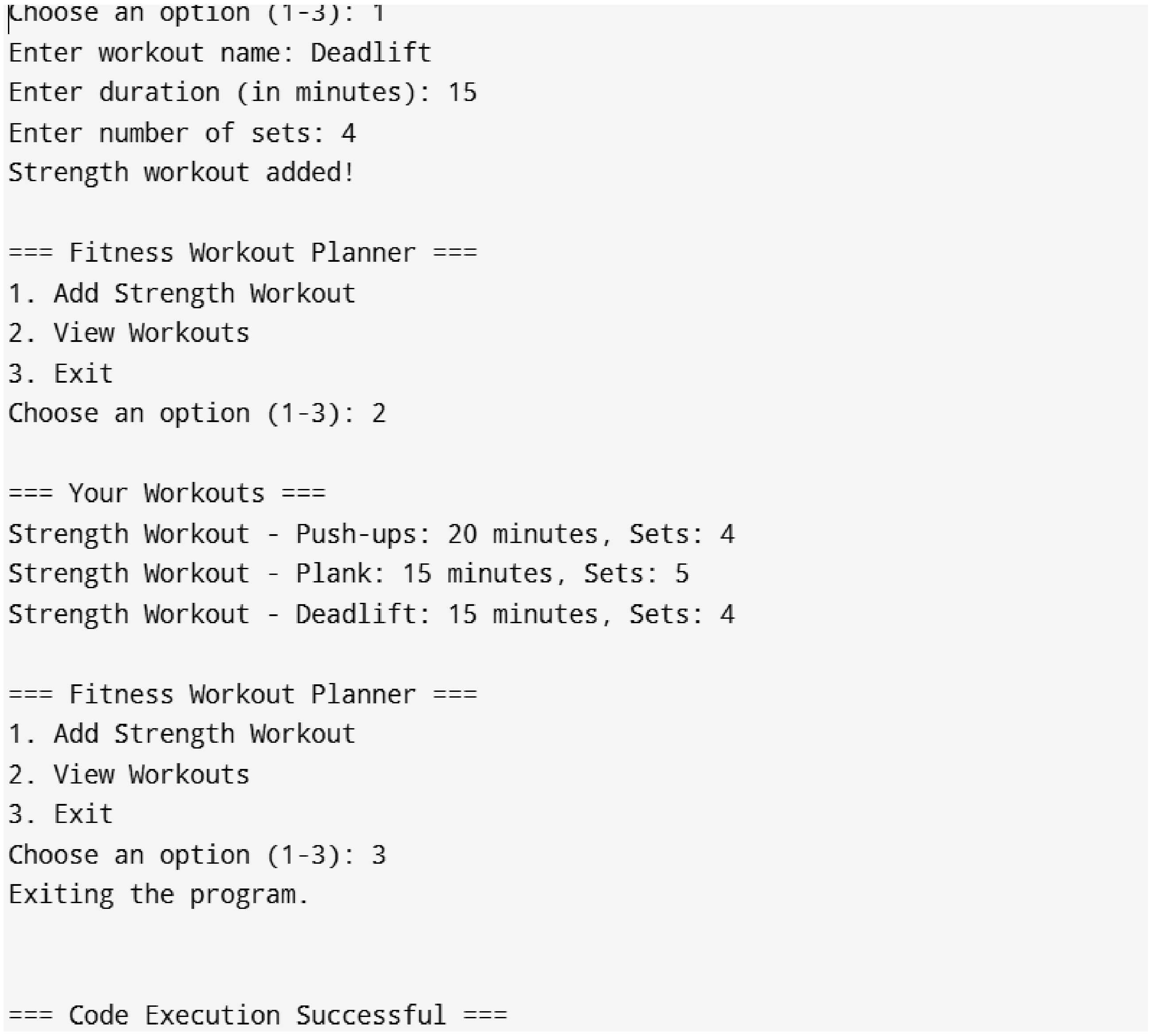
```
#include<iostream>

#include <string>

using namespace std;

const int MAX_WORKOUTS = 10;

class Workout {

protected:

   string name;

   int duration; // in minutes

public:

   Workout(const string& name, int duration) : name(name), duration(duration) {}

   virtual void display() const {

       cout << name << ": " << duration << " minutes\n";

   }

};

class Strength : public Workout {

private:

   int sets; // number of sets


public:

   Strength(const string& name, int duration, int sets)

       : Workout(name, duration), sets(sets) {}


   void display() const override {

       cout << "Strength Workout - " << name << ": " << duration << " minutes, Sets: " << sets <<
"\n";

   }

};
```

```
void displayMenu() {
    cout << "\n=== Fitness Workout Planner ===\n";
    cout << "1. Add Strength Workout\n";
    cout << "2. View Workouts\n";
    cout << "3. Exit\n";
    cout << "Choose an option (1-3): ";
}
int main() {
    Workout* workouts[MAX_WORKOUTS]; // Array of pointers to Workout
    int workoutCount = 0; // Counter for workouts
    int choice;
    do {
        displayMenu();
        cin >> choice;

        if (choice == 1) {
            if (workoutCount < MAX_WORKOUTS) {
                string name;
                int duration, sets;
                cout << "Enter workout name: ";
                cin.ignore();
                cin>> name;
                cout << "Enter duration (in minutes): ";
                cin >> duration;
                cout << "Enter number of sets: ";
                cin >> sets;
                workouts[workoutCount++] = new Strength(name, duration, sets);
                cout << "Strength workout added!\n";
            } else {
                cout << "Maximum workouts reached.\n";
            }
        } else if (choice == 2) {
            cout << "\n=== Your Workouts ===\n";
            for (int i = 0; i < workoutCount; i++) {
                workouts[i]->display();
            }
        } else if (choice == 3) {
            cout << "Exiting the program.\n";
        } else {
            cout << "Invalid choice. Please enter a number between 1 and 3.\n";
        }
    } while (choice!= 3);

    // Clean up allocated memory
    for (int i = 0; i < workoutCount; i++) {
        delete workouts[i];
    }
    return 0 ;
}
```

Output



```
=== Fitness Workout Planner ===
1. Add Strength Workout
2. View Workouts
3. Exit
Choose an option (1-3): 1
Enter workout name: Push-ups
Enter duration (in minutes): 20
Enter number of sets: 4
Strength workout added!

=== Fitness Workout Planner ===
1. Add Strength Workout
2. View Workouts
3. Exit
Choose an option (1-3): 1
Enter workout name: Plank
Enter duration (in minutes): 15
Enter number of sets: 5
Strength workout added!

=== Fitness Workout Planner ===
1. Add Strength Workout
2. View Workouts
3. Exit
```



```
Choose an option (1-3): 1
Enter workout name: Deadlift
Enter duration (in minutes): 15
Enter number of sets: 4
Strength workout added!

=== Fitness Workout Planner ===
1. Add Strength Workout
2. View Workouts
3. Exit
Choose an option (1-3): 2

=== Your Workouts ===
Strength Workout - Push-ups: 20 minutes, Sets: 4
Strength Workout - Plank: 15 minutes, Sets: 5
Strength Workout - Deadlift: 15 minutes, Sets: 4

=== Fitness Workout Planner ===
1. Add Strength Workout
2. View Workouts
3. Exit
Choose an option (1-3): 3
Exiting the program.


=== Code Execution Successful ===
```

Conclusion

This fitness workout planner is a solid starting point for managing workouts effectively. It demonstrates how to use classes and handle user input in a straightforward way. The ability to add and view strength workouts makes it useful for users looking to track their fitness progress.

To enhance the program, consider adding more workout types, such as cardio, yoga, or flexibility exercises. This would give users a wider range of options to choose from, making the app more versatile. Implementing a feature to save workouts to a file would also help users keep their data safe between sessions, allowing for easier access late.Improving error checking and input validation would make the program more reliable and user-friendly. For example, ensuring that users enter valid numbers for duration and sets would reduce mistakes. Additionally, adding options to edit or delete existing workouts would improve the overall user experience.

Overall it is used for building a more complete fitness application. With a few updates and enhancements, it could become a valuable tool for fitness enthusiasts at any level. By focusing on user needs and expanding features, the program could significantly contribute to achieving fitness goals.